# Assignment #2
## : Final report

학번 : 2021320450
학과 : 컴퓨터학부
이름 : 박성준

---

## 소개

본 리포트는 대표적인 이미지 데이터 셋 중 하나인 CIFAR-10을 사용하여 Image Classification을 수행과정을 보고하기 위해 작성되었다.

훈련 데이터로부터 특징을 학습하는 딥 러닝 알고리즘들은 좋은 성능을 나타내기 위해 데이터 개수가 매우 중요하다. 부족한 데이터 문제를 해결하기 위해 사전 학습 모델(Pretrained model)을 사용하는 것은 많은 데이터를 학습한 모델의 지식을 전달받을 수 있어 타겟 데이터 셋에 최적화가 빠르고 일반화 성능이 높아지는 장점이 있다.

하지만 사전 학습 모델을 사용하는 것이 항상 잘 동작하는 것은 아니다. 이유는 다음과 같다. 1. 학습된 데이터 셋의 분포와 타겟 데이터 셋의 분포가 비슷해야 사전 학습의 이득을 취할 수 있다. 만약 ImageNet으로 사전 학습된 모델을 사용할 경우 타겟 도메인이 ImageNet과 비슷한 분포를 가지면 객체에 대한 특징을 잘 추출할 수 있지만, 타겟 도메인이 3D 의료 영상일 경우 데이터의 분포 차이로 인해 오히려 모델 최적화에 방해가 될 수 있다. 2.사전 학습 이미지의 크기와 타겟 도메인의 입력 이미지의 크기를 고려하는 것이 중요하다. 데이터 크기에 따라 적용되는 소스 도메인 데이터와 타겟 데이터 셋에 적절한 Convolutional filter를 적용시키는 것이 빠른 Feature Extraction에 중요하기 때문이다.

본 리포트에서는 ImageNet Pretrained model을 효과적으로 사용하기 위해 시도한다. 소스 도메인은 ImageNet 데이터 셋, 타겟 도메인은 CIFAR-10 데이터 셋이 사용된다. 여기서 ImageNet은 224x224x3 크기로 훈련되었고, CiFAR-10은 32x32x3 크기로 훈련된다. 여기서 큰 영상을 학습한 ImageNet 모델에 CIFAR-10을 fine-tuning하게 되면 소스 도메인과 다른 크기로 학습하게 되어 효율적인 학습을 할 수 없다. 이를 해결하기 위해 CIFAR-10 이미지를 보간(ex, Bilinear, Bicubic Interpolation) 알고리즘으로 크기를 키워서 ImageNet 크기로 맞춰 학습하는 방법이 있다. 하지만 단순 보간 알고리즘은 이미지가 흐려져 훈련에 데이터의 특징이 사라질 수 있다.

본인은 영상 크기를 조절할 때 단순한 보간 알고리즘을 사용하는 것이 아닌 Resizer Network[1]를 사용한다. Resizer 모델은 입력 영상을 손실 함수를 최적화하도록 학습된다. 본인은 Resizer을 효율적으로 사용하기 위해 Resizer 모델의 웨이트에 중요도를 부여하고 채널, 공간 사이에 Attention Block을 적용해 학습한다. 그 결과 작은 영상에서 큰 영상으로 보간 중 흐릿한 부분을 완화시키고

중요한 부분을 강조하여 향상된 분류 성능을 보였다.

## 학습 전략

본 과제에서 타겟 도메인 CIFAR-10은 32x32x3크기로 비교적 작은 편이다. 입력 영상의 Data complexity가 크지 않기 때문에 백본 네트워크는 Model complexity가 비교적 작은 VGG16[4]을 사용하였다.

Resizer는 백본 네트워크(VGG[4])의 앞에 결합되어 End-to-end로 학습된다. Resizer의 구조는 그림2와 같다.

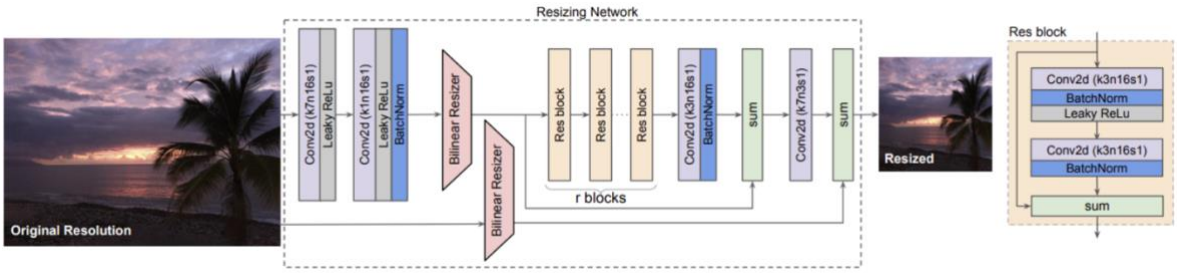


그림 1) A Learnable Resizer Module[1] 구조도

Resizer (Network)는 학습 파라미터로 생기는 Feature map과 Interpolation의 결합으로 구성된다. Resizer의 학습 파라미터는 Conv Block과 Res Block으로 구성되어 있어 손실 함수를 최적화하는 과정에서 효율적인 변환을 자동으로 학습한다.

하지만 입력 영상이 Resizer에 의해 변환되는 영상이 항상 도움을 주지 않을 수 있다. 따라서 Resizer모델 파라미터를 효율적으로 사용하기위해 Resizer의 파라미터 끝에 CBAM(Convolutional Block Attention Module)[2]을 추가하여 Channel, Spatial 단위에 강조할 수 있도록 시도한다. CBAM의 구조도는 그림 3과 같다.

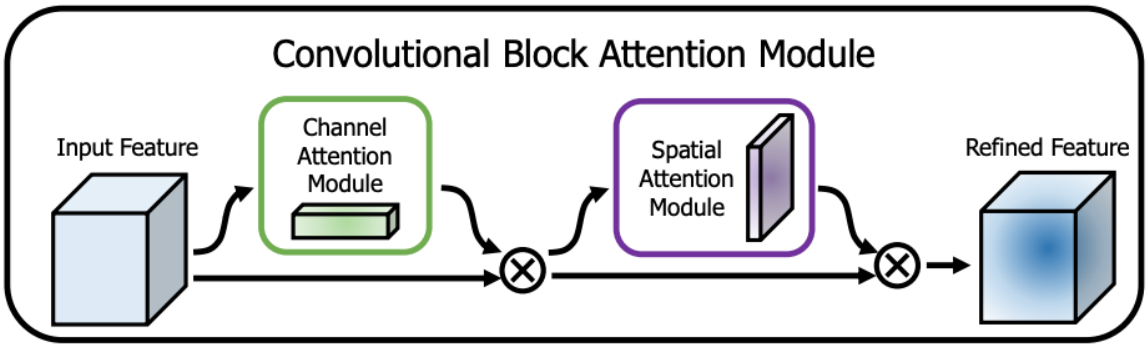


그림 2) Convolutional Block Attention Module[2] 구조도

CBAM은 Input feature를 입력으로 Channel, Spatial 단위로 Attention mechanism을 수행한다. 이로 인해 피쳐의 중요한 부분에 더 가중치를 부여하게 된다. 본 과제에서는 Resizer의 Resblock과 CBAM을 결합하여 Resizer_CBAM을 사용한다. ResBlock과 CBAM의 결합 구조는 그림 4와 같이 구성된다.

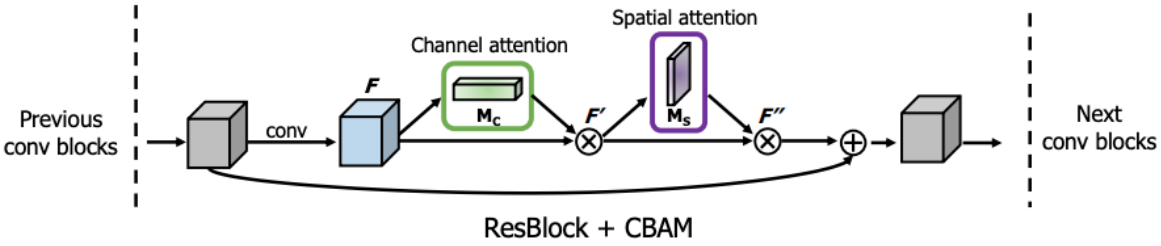


그림 3) Convolutional Block Attention Module[2] 구조도

Resizer와 CBAM을 결합하여 사용하여 과 같이 CBAM은 입력 피쳐를 Channel, Spatial 단위로 어텐션을 수행하여 피쳐맵을 강조할 수 있다. Resizer와 CBAM와 결합하여 Resize하는 과정에서 학습을 통해 흐릿한 부분에 대한 필요한 부분을 강조하여 효과적인 성능 향상을 보인다.

## 실험 방법

데이터 셋

CIFAR-10 데이터 셋은 10개 클래스로 각 클래스당 6,000장으로 구성되어 있으며 32x32 컬러 이미지로 구성되어 있다. 총 60,000장의 데이터 셋 중 50,000개의 훈련 이미지와 10,000개의 테스트 이미지로 구성되어 있다. 다음은 데이터 셋의 클래스와 각 10개의 임의로 선택된 이미지의 예시를 보여준다.

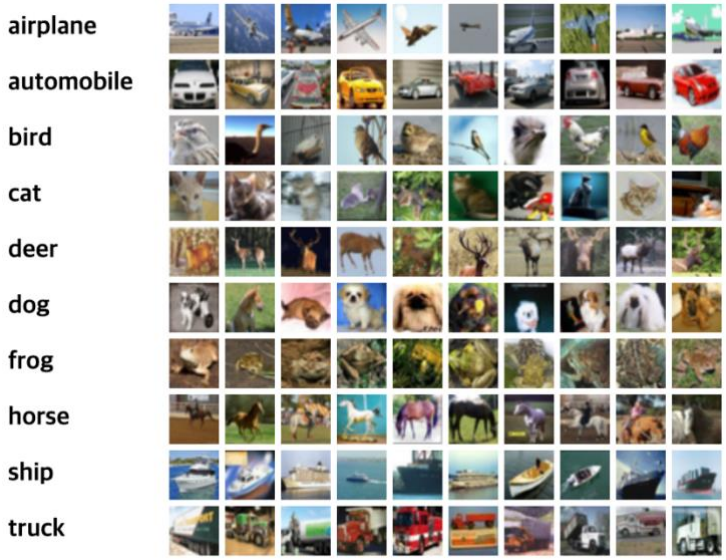


그림 4) CIFAR-10 데이터 셋 예제

데이터 전처리

학습 데이터는 32x32x3의 크기로 구성되어 있다. 학습 중에 영상은 padding 4를 유지한체 RandomCrop,을 수행 후, 50%의 확률로 RandomHorizontalFlip 후, ImageNet normalization을 수행하였다. 모든 Augmentation 기법은 Torchvision 패키지를 사용하였다.

```
transform_train = transforms.Compose([
    transforms.RandomCrop(32, padding=4),                # Random Position Crop
    transforms.RandomHorizontalFlip(),                   # right and left flip
    transforms.ToTensor(),                               # change [0,255] Int value to [0,1] Float value
    transforms.Normalize(mean=(0.4914, 0.4824, 0.4467), # RGB Normalize MEAN
                         std=(0.2471, 0.2436, 0.2616))  # RGB Normalize Standard Deviation
])
```

그림 5) 훈련 데이터 셋 데이터 전처리 방법

```
transform_test = transforms.Compose([
    transforms.ToTensor(),                               # change [0,255] Int value to [0,1] Float value
    transforms.Normalize(mean=(0.4914, 0.4824, 0.4467), # RGB Normalize MEAN
                         std=(0.2471, 0.2436, 0.2616))  # RGB Normalize Standard Deviation
])
```

그림 6) 테스트 데이터 셋 데이터 전처리 방법

훈련 과정에서 그림 5의 전처리가 끝난 후 50%의 확률로 Cutmix[3] 기법을 수행하여 일반화 성능을 향상시켰다. 그림 7)은 훈련 과정에서 사용된 Cutmix결과를 랜덤 추출한 것 이다.

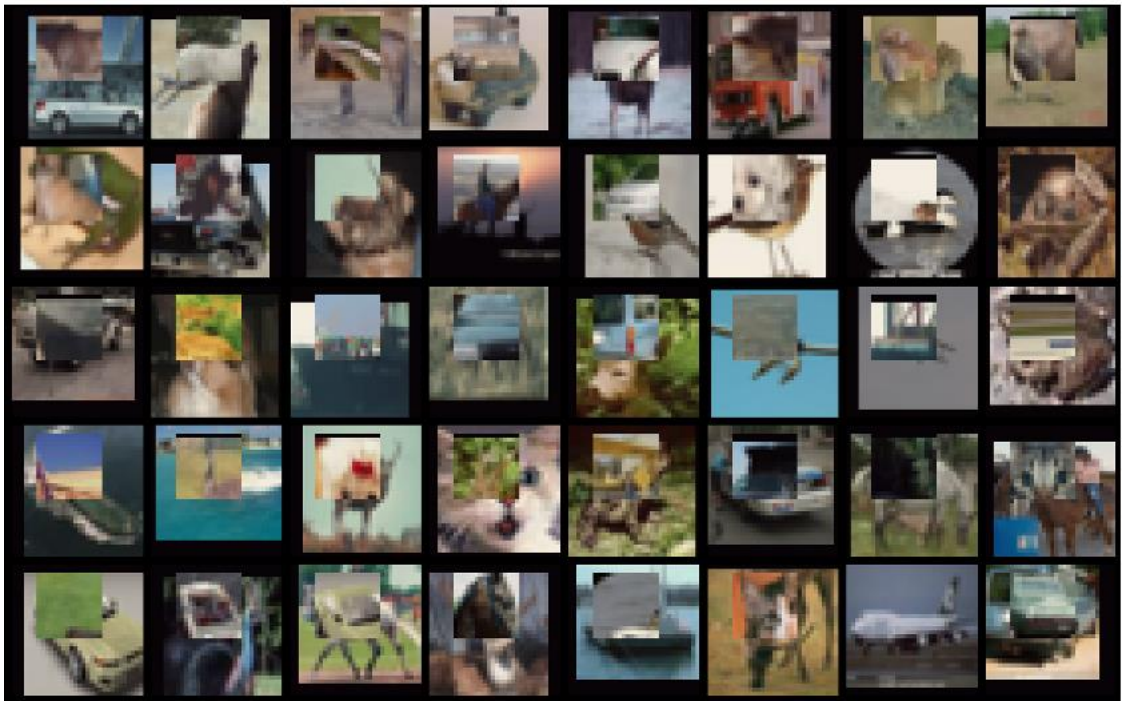

그림 7) CIFAR-10에 적용된 Cutmix 결과 예제

모델 세팅

본 과제에서 백본 모델은 Pytorch의 Torchvision 패키지의 ImageNet pretrained VGG16을 사용하고 Classifier를 cifar-10에 맞게 10개의 클레스를 예측하도록 수정하였다. 모델은 batch size 128, learning rate 0.01에서 300에폭만큼 SGD 옵티마이저를 사용하여 훈련되었다. 100, 150 epochs에서 Learning rate를 0.1씩 줄였다. Resizer는 32x32x3 크기의 원본 이미지를 224x224x3으로 변환하였다. 모델은 300에폭 훈련 중 가장 높은 정확도를 가지는 모델을 선택하였다. 학습 과정에서 Tesla V100 GPU를 사용했다.

## 실험 결과

본 과제에서 CIFAR-10에 대한 성능은 표 1)와 같다. 본 연구에서는 VGG-16, Resizer_CBAM, Cutmix augmentation을 수행하여 최종적으로 96.63%의 정확도를 달성했다.

| Model (VGG[4]) | Baseline | Baseline + Cutmix | Baseline + Resizer | Baseline + Resizer + Cutmix | Baseline + Resizer_CBAM | Baseline + Resizer_CBAM + Cutmix |
|---|---|---|---|---|---|---|
| Accuracy | 93.13 | 94.84 | 95.68 | 96.61 | 96.09 | **96.63** |

표 1) CIFAR-10에 대해 VGG16 모델 성능 비교 결과

또한 그림 8은 영상을 32x32x3에서 224x224x3크기로 변화시킬 때 Bilinear interpolation 알고리즘, Resizer_CBAM, 두 영상의 차이를 시각적으로 나타낸 결과이다.

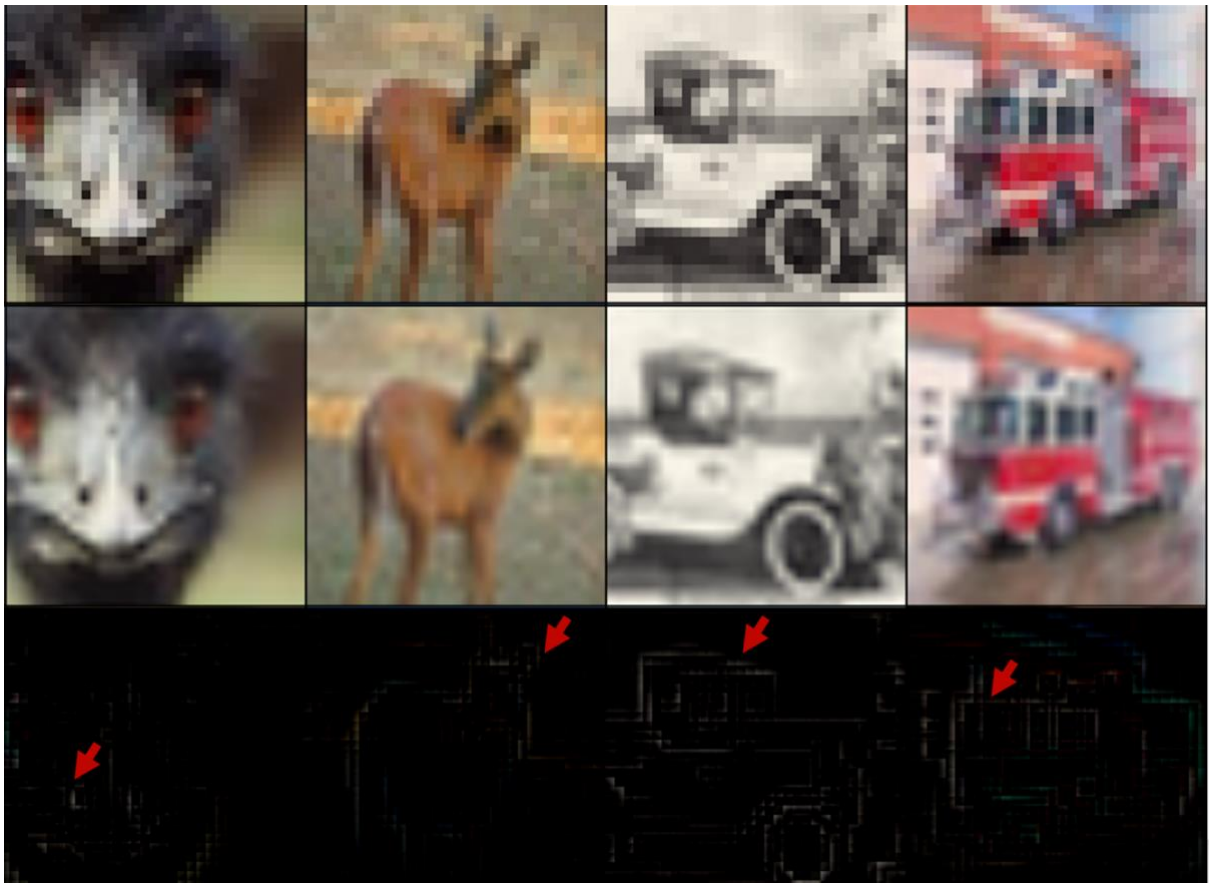

그림 8) 32x32 크기 영상을 224x224로 변환했을 때 결과
Row : 가장 위부터 Bilinear Interpolation, Resizer_CBAM, 두 영상 차이,
Column : CIFAR-10 테스트 셋 영상

Bilinear Interpolation의 경우 224x224x3 크기로 변환 시킬 때 영상이 흐릿해지는 반면, Resizer를 사용할 때 학습 과정 중 좀더 선명하게 학습이 되는 모습을 보여준다. 여기서 주의할 점은 마지막 차이 영상에서 여기서 물체를 쉽게 구분하기 위해 Resizer가 물체의 엣지를 좀더 강조하는 경

향을 보인다.

## 결론

본 과제에서는 Resizer와 CBAM을 결합하여 효율적으로 모델 크기를 조절했다. CIFAR-10 데이터 셋에서 pretrained VGG 네트워크를 사용해서 기존 성능 대비 3.5% 정확도 향상을 보였다. 앙상블, Test Time Augmentation을 수행하여 더욱 성능이 향상될 것을 기대한다.

참고 문헌

[1] Learning To Resize Images for Computer Vision Tasks, Talebi_2021_ICCV
[2] CBAM: Convolutional Block Attention Module, Sanghyun Woo_2018_ECCV
[3] CutMix: Regularization Strategy to Train Strong Classifiers with Localizable Features, Sandoo Yun_2019_ICCV
[4] Very Deep Convolutional Networks for Large-Scale Image Recognition, Simonyan_Arxiv_2014